Supplementary: Separating Style and Content for Generalized Style Transfer

Yexun Zhang
Shanghai Jiao Tong University
zhyxun@sjtu.edu.cn

Ya Zhang[✉]
Shanghai Jiao Tong University
ya_zhang@sjtu.edu.cn

Wenbin Cai
Microsoft
wenbca@microsoft.com

In this supplementary, we present more experimental results to better validate the effectiveness of our proposed method. We first conduct experiments to perform morphing between two styles. Then, we give more experimental results for factors influencing the model performance and present both the quantitative and qualitative results. Finally, we present some results for neural style transfer.

1. Morphing

In this subsection, we perform morphing between two styles. We synthesize new styles by changing the weight between two styles S_1 and S_2 according to the following function:

$$S_{\text{New}} = (1-\lambda) \times S_1 + \lambda \times S_2, \qquad 0 \leq \lambda \leq 1. \tag{1}$$

The styles and contents used in this experiment are all novel. During experiment, we first extract the style features for the two styles from style reference sets $\mathcal{R}_{S_1}$ and $\mathcal{R}_{S_2}$ and then combine them with different weight λ. Finally, the new style feature S_{New} will be combined with the content feature and generate the image. The results are presented in Figure 1 and Figure 2. From the figure, we can observe the changing process from style S_1 to style S_2. This experiment further validates that the *Style Encoder* can extract accurate and pure style features. Besides, by separating style and content, we can leverage the style representations to create new styles.

Figure 1: Results of morphing between two styles. $\mathcal{R}_{S_1}$: Reference set for style S_1, $\mathcal{R}_{S_2}$: Reference set for style S_2, TG1: Target images for style S_1, TG2: Target images for style S_2, 0.0-1.0: Outputs for λ = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0].

Figure 2: Results of morphing between two styles. $\mathcal{R}_{S_1}$: Reference set for style S_1, $\mathcal{R}_{S_2}$: Reference set for style S_2, TG1: Target images for style S_1, TG2: Target images for style S_2, 0.0-1.0: Outputs for λ = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0].

2. Influence of the Training Set Size

In this section, we present the quantitative results of different training set size in Table 1 and the qualitative results in Figure 3. We can observe that for both quantitative results and qualitative results, the larger the training set size, the better the performance. In addition, the model performance saturates with the increase of the training set size.

Table 1: Quantitative comparison of models with different training set size.

	D_1			D_2			D_3			D_4		
	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR
20k	0.0096	0.0192	0.1801	0.0096	0.0192	0.1806	0.0095	0.0191	0.1758	0.0095	0.0191	0.1764
50k	0.0096	0.0191	0.1713	0.0097	0.0192	0.1726	0.0095	0.0191	0.1668	0.0096	0.0192	0.1679
100k	0.0093	0.0188	0.1662	**0.0094**	**0.0189**	0.1686	0.0093	0.0188	0.1633	**0.0094**	**0.0189**	0.1654
300k	**0.0091**	**0.0185**	0.1549	**0.0094**	**0.0189**	0.1604	**0.0092**	**0.0187**	0.1549	**0.0094**	**0.0189**	0.1592
500k	**0.0091**	**0.0185**	**0.1509**	**0.0094**	**0.0189**	**0.1578**	**0.0092**	**0.0187**	**0.1519**	0.0095	0.019	**0.1569**

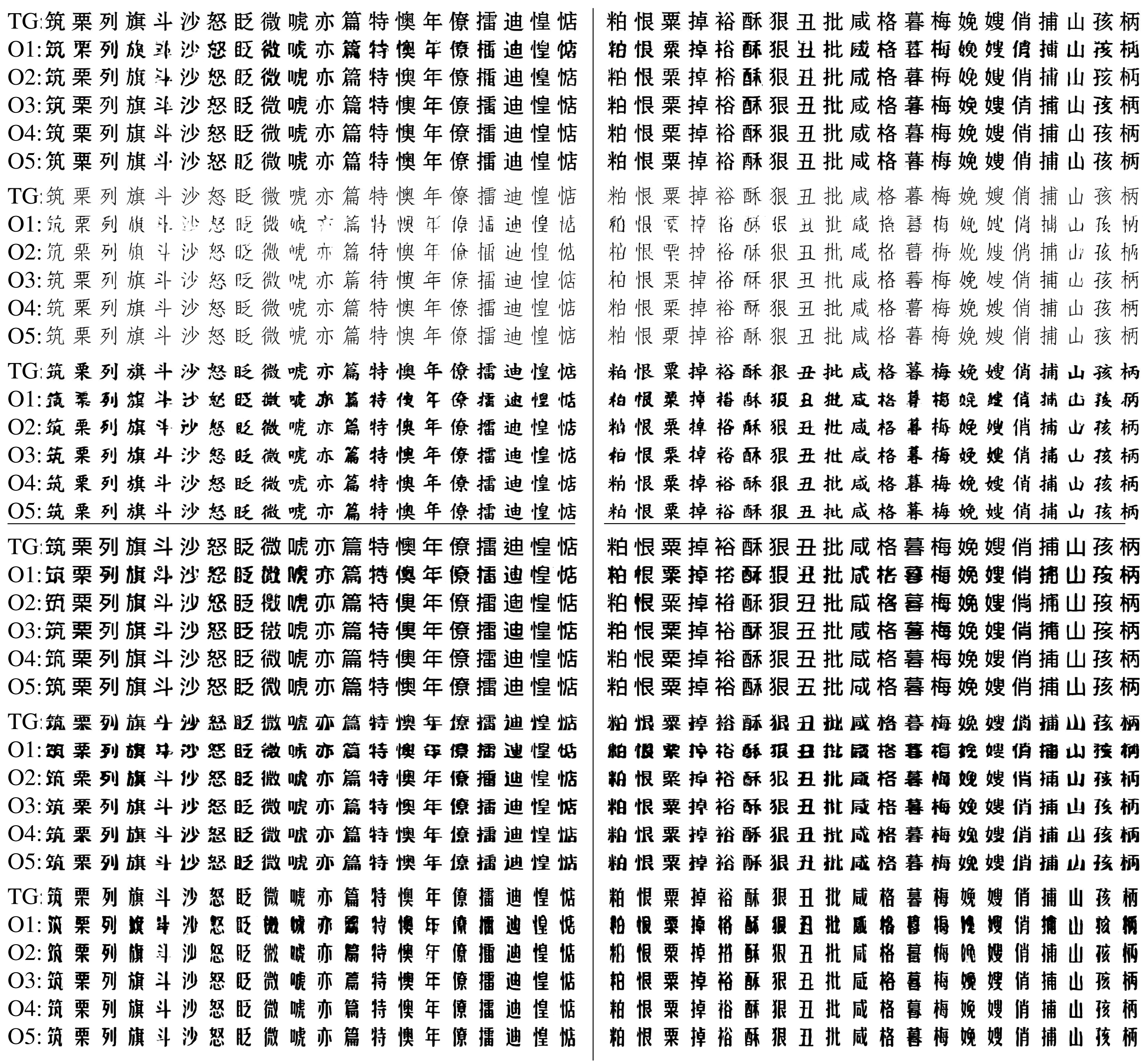

Figure 3: Generation for D_1, D_2, D_3, D_4 (from upper left to lower right) with different training set size. TG: Target image, O1: Output for N_t=20k, O2: Output for N_t=50k, O3: Output for N_t=100k, O4: Output for N_t=300k, O5: Output for N_t=500k. In all cases, r=10.

3. Influence of Reference Set Size

Following, we present the quantitative results of different reference set size in Table 2 and more generated images in Figure 4. From the figure, we can observe that r=2 performs worst and r=10 and r=15 perform closely, indicating that more reference images will provide more information and the performance will be saturated with the increase of reference set size.

Table 2: Quantitative comparison of models with different reference set size. In all cases, N_t=300k.

	D_1			D_2			D_3			D_4		
	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR
r=2	0.0096	0.0191	0.1635	0.0098	0.0193	0.1677	0.0097	0.0192	0.1611	0.0098	0.0193	0.1649
r=5	0.0093	0.0188	0.1594	0.0095	0.019	0.1641	0.0094	0.0189	0.1578	0.0096	0.0192	0.1615
r=10	0.0091	0.0185	0.1549	0.0094	0.0189	0.1604	0.0092	0.0187	0.1549	0.0094	0.0189	0.1592
r=15	0.0091	0.0186	0.1557	0.0094	0.0189	0.1601	0.0092	0.0187	0.1552	0.0095	0.019	0.1584
r=50	**0.009**	**0.0184**	**0.1533**	**0.0092**	**0.0187**	**0.1585**	**0.0091**	**0.0185**	**0.1537**	**0.0093**	**0.0188**	**0.1571**

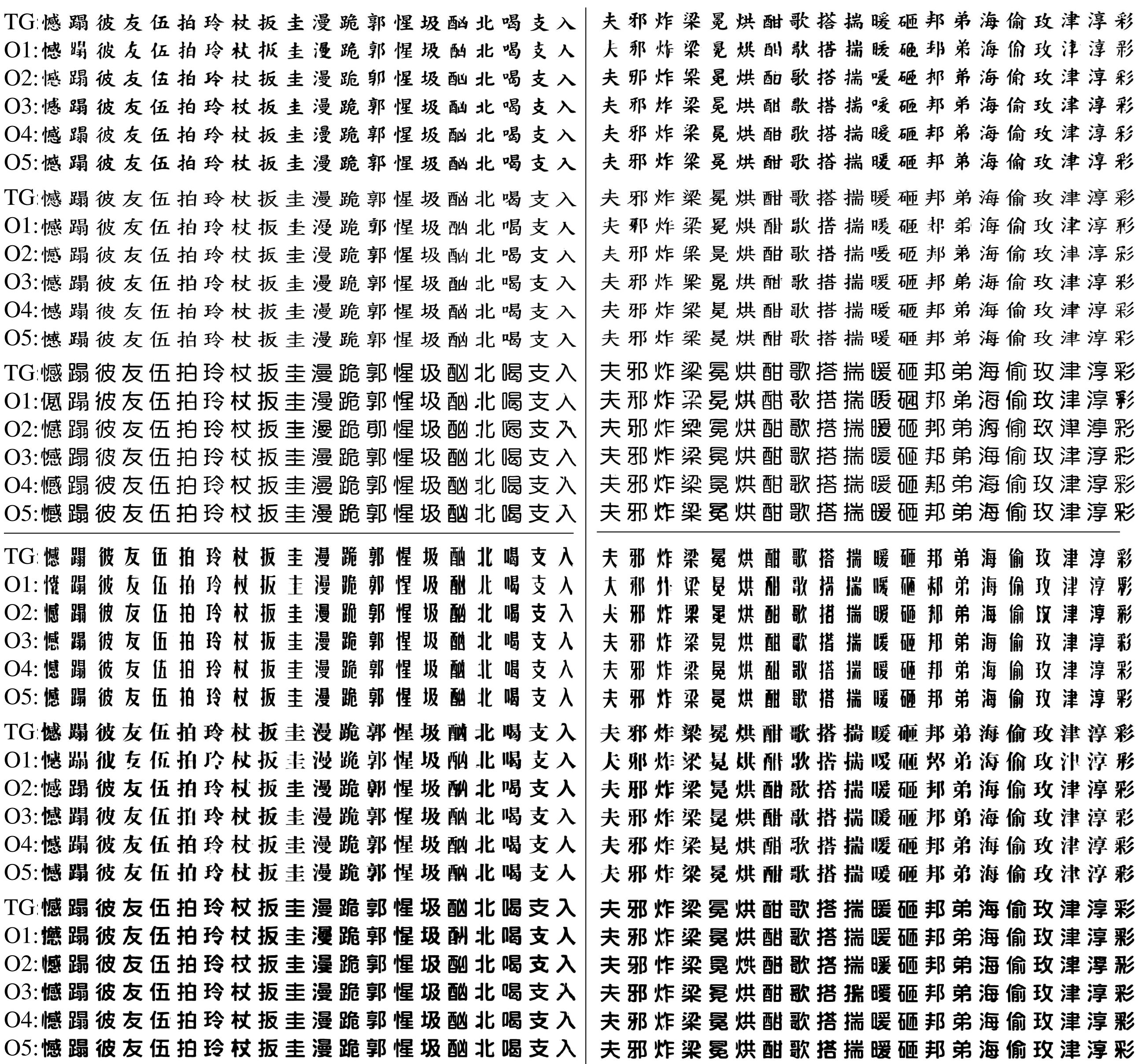

Figure 4: The impact of the number of reference images on the generation of images in D_1, D_2, D_3, D_4, respectively (from upper left to lower right). TG: Target image, O1: Output for r=2, O2: Output for r=5, O3: Output for r=10, O4: Output for r=15 and O5: Output for r=50. In all cases, N_t=300k.

4. Effect of the Weighted Loss

In this subsection, we compare the model trained with L1 loss and weighted L1 loss. The quantitative results are displayed in Table 3 and the qualitative results are shown in Figure 5. From the figure, we can observe that images with thin and light characters are generated better with weighted loss.

Table 3: Quantitative comparison of models with L1 loss and weighted L1 loss.

	D_1			D_2			D_3			D_4		
	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR	L1 loss	RMSE	PDAR
L1 loss	**0.0091**	0.0186	0.1561	**0.0094**	**0.0189**	0.161	0.0093	**0.0187**	0.1554	0.0095	0.019	**0.1592**
Weighted L1 loss	**0.0091**	**0.0185**	**0.1549**	**0.0094**	**0.0189**	**0.1604**	**0.0092**	**0.0187**	**0.1549**	**0.0094**	**0.0189**	**0.1592**

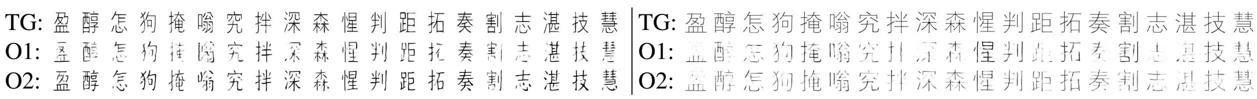

Figure 5: Results of different loss functions with N_t=300k, r=10. TG: Target image, O1: Output for L1 as the loss function, O2: Output for weighted L1 loss as the loss function.

5. Results of one reference image

We compare two models: r=10 vs. r=1 (splitting each former triplet into 100 triplets). As shown in Figure 6, the two models perform similarly, but the first model is more time efficient since it learns from r^2 style-content pairs at one time.

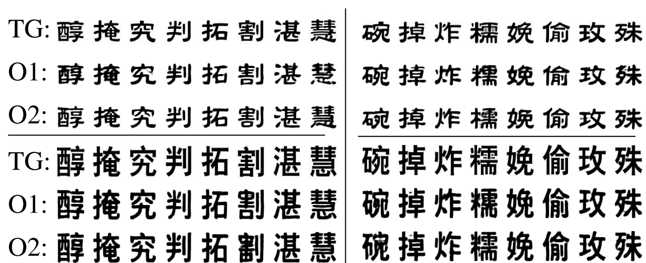

Figure 6: Generation for D_1, D_2, D_3, D_4 (from upper left to lower right) for N_t=300k. TG: Target image, O1: Output for r=10, O2: Output for r=1.

6. Experiment for neural style transfer

For neural style transfer, we constructed a dataset with artistic Photoshop filters which contains 106 styles and each style has 781 images with different contents. The results on the dataset are presented in Figure 7, showing our method works well for neural images.

Figure 7: Experiment results for neural style transfer.